inotify

Fermi Linux Users Group
August 29 2012

How did we monitor filesystem "events"?

...Polling?

How did we monitor filesystem "events"?

...Polling?

- "Busy" polling processes can add up

How did we monitor filesystem "events"?

...Polling?

- "Busy" polling processes can add up
- Race between files added/deleted and reread

inotify

- An API used to monitor changes to a filesystem.

inotify

- An API used to monitor changes to a filesystem.

- Available in kernels 2.6.13 or later.

inotify

- An API used to monitor changes to a filesystem.
- Available in kernels 2.6.13 or later.
- Replaces `dnotify`

System Calls

- `inotify_init()`
  Instantiates subsystem in the kernel

- `inotify_add_watch()`
  Creates a watch; takes a pathname and event(s) , returns identifier

- `inotify_rm_watch`
  Takes identifier, removes watch

inotify-tools

- Available from epel.

- Provides two commandline programs:, `inotifywatch` and `inotifywait`.

inotifywait

Uses the inotify interface to wait for changes to files.

```
   -r, --recursive
   -m, --monitor (otherwise exit on
event)
   -s, --syslog
   -o, --outfile

# inotifywait -r -m /home/bonniek/foo
```

inotifywait

-e <event> access|modify|close|open|move|create|delete|unmount ...

-@<file> (exclude file or directory)

--format <fmt> (printf-like syntax):

- %w file name
- %e events
- %T time as (can be specified with --timefmt)

inotify example (from man page)

```
#!/bin/sh
while inotifywait -e modify /var/log/messages; do
       if tail -n1 /var/log/messages | grep httpd; then
           # Do something!  Maybe this (probably not this).
           kdialog --msgbox "Apache needs love!"
       fi
done
```

inotifywatch

Outputs a table showing number of times an event occured for each watched file or dir.

```
   -t, --timeout <seconds> (otherwise
exit on signal)
   -r, --recursive
   -e <event>, --event <event>
   --fromfile <file> (read filenames to
watch from file)
```

inotifywatch

```
[bonniek@gerda]$ inotifywatch -v -e access -e modify -e open -e
close -r -t 30 /var/log/

Establishing watches...
Setting up watch(es) on /var/log/
OK, /var/log/ is now being watched.
Total of 17 watches.
Finished establishing watches, now collecting statistics.
Will listen for events for 30 seconds.
total   modify  close_write    close_nowrite  open  filename
21      3       3              6              9       /var/log/
12      12      0              0              0   /var/log/httpd/
```

inotifywatch: more options

-z, --zero (output table rows even if count is zero)

@<file> (exclude file)

--exclude <pattern> (exclude files matching <pattern> regex)

--excludei <pattern> (like --exclude but case-insensitive)

lsyncd

Available from epel, uses inotify to monitor for evens, then spawns process(es) to sync (usually rsync).

```
# lsyncd [OPTIONS] -rsyncssh SOURCEDIR
TARGETHOST TARGETDIR
```

...for local file ops:

```
# lsyncd [OPTIONS] -direct SOURCEDIR
TARGETDIR
```

incron

Like cron, but schedule jobs according to filesystem events.

WIth `incrond` running:

```
# incrontab -e file
```

incron

```
/home/bonniek/foo IN_DELETE,IN_MOVE touch $@/$#
/home IN_CREATE mail -s "Welcome to Fermilab" $#@fnal.gov <
/some/mail.txt
```

Wildcards:

$$ dollar sign

$@ watched filesystem path (see above)

$# event-related file name

$% event flags (textually)

$& event flags (numerically)

incron

Events:

IN_ACCESS File was accessed (read) (*)

IN_ATTRIB Metadata changed (permissions, timestamps, extended attributes, etc.) (*)

IN_CLOSE_WRITE File opened for writing was closed (*)

IN_CLOSE_NOWRITE File not opened for writing was closed (*)

IN_CREATE File/directory created in watched directory (*)

IN_DELETE File/directory deleted from watched directory (*)

IN_DELETE_SELF Watched file/directory was itself deleted

IN_MODIFY File was modified (*)

IN_MOVE_SELF Watched file/directory was itself moved

IN_MOVED_FROM File moved out of watched directory (*)

IN_MOVED_TO File moved into watched directory (*)

IN_OPEN File was opened (*)

fsnotify

Better performance on whole filesystems.

New in 2.6.31

`inotify` (and `dnotify`) are re-implemented on top.

more info

https://github.com/rvoicilas/inotify-tools/wiki/

http://www.ibm.com/developerworks/linux/library/l-ubuntu-inotify/index.html

http://lwn.net/Articles/311850/